EEL 4930/5934 BioSignals Processing

Assignment 2- Due: October 12

Luiz Medeiros

Please write a report answering the following questions. Submit the report and all the m-files you used to answer the questions on Blackboard in Module Assignments / Assignment 2.

1. Exercises with known functions: Generate a discrete-time signal  sampled from a sinusoid of frequency 1 Hz. at the rate of  and duration of  seconds. We will refer to the signal as the MATLAB variable ***x***.f
   1. Using the ***subplot*** option, ***plot*** ***x*** versus *time* in seconds for seconds for . You should have one figure for each sampling rate.  
        
        
      

%%Exercise 1

clc,clear all,close all;

%a.

Td=[1 2 4 8 16]; % final sampling time in seconds;

sr=[4 8 16]; % sampling rate;

f0=1000;

Fs=[];

Ts=[];

t=[]; % will serve to store our final time frames based on the

% above evaluations;

x=[];

strTd='';

strTs='';

strF='';

lt=[];

for k=1:5

strTd=num2str(Td(k));

figure(k);

for n=1:3

Fs(n)=sr(n)\*f0;

Ts(n)=1/Fs(n);

t=0:Ts(n):Td(k)-Ts(n);

x=sin(2\*pi\*f0\*t);

strTs=num2str(Ts(n));

strF=strcat('Td= ',strTd,',','Ts= ',strTs);

lt=size(find(t<0.003)); % This will provide me all the samples before 3 ms.

subplot(3,1,n),plot(t(1,1:lt(1,2)),x(1,1:lt(1,2))),title(strF); % only outputting the samples up to 3ms.

end

end

* + 1. Are the sampling rates adequate to reconstruct the sinusoid of 1 Hz.?

According to Nyquist Sampling Theorem, we should sample the signal at 2\*f0, or 2 times its fundamental frequency. In our case, we are beginning our sampling at 2n\*f0= 4\*f0=4\*1000, where n begins at 2.Thereafter we increase n up to 4 . In theory, these are good to very good sampling rates.

* + 1. Are the sampling rates adequate to draw a smooth plot of the sinusoid of 1 Hz.?

The most adequate sampling rate to draw a smooth sinusoid is 16\*f0 = 16000.

* + 1. Explain the difference.

The difference is the amount of samples we obtain per period. Take the figure below as an example (which displays a little more than two periods of the sampled signal). It contains 4 samples per period, then 8 and then 16. Consequently, as seen on the picture, we see a much smoother picture with the larger number of samples. In addition, with each increased number of samples, we decrease the probability that there may be an aggregate function which the picture is not displaying. So our sampled signal becomes more and more accurate.



* 1. Find ***fx***, the ***nfft*** *-point* DFT of ***x***, for , ***nfft***=512, .
     1. Using the ***subplot*** option, ***plot*** the magnitude of ***fx*** versus true frequency 0- Hz. for each value of . This exercise tests the effect of signal duration on the DFT.

Below is the result of our DFT image.



* + 1. Comment on the results and draw a conclusion from your observation.

The results in itself were as expected. We over sampling by 4 times our Nyquist rate (2\*f0 vs 8\*f0). Moreover, the longer duration of the signal only gives for the probability that further signals may interfere and thereby aggregate to our signal of interest. If this had happened, say a random signal at 60Hz, we would see peaks delineating these harmonics and their respective strength on the graph. Since we had a strictly controlled signal, for all durations, we only perceived one strong peak at the fundamental frequency of 1000Hz and very weak subsequent peaks at the latter harmonics (for example at 2000 and 3000Hz).

In addition, the longer the signal we retain, the more energy we retain, as each peak signify energy levels.

* 1. Compute ***fx*** for , , ***nfft***,
     1. Using the ***subplot*** option, ***plot*** the magnitude of ***fx*** versus true frequency 0- Hz. for each value of ***nfft***. This exercise tests the effect of ***nfft*** on the appearance of the DFT.

Paste the figure and the MATLAB code here.  


Through these exercises we are able to see the effect of the different points of the DFT, FFT algorithm, and the related results.

* + 1. Comment on the results.

Essentially, as the nfft increases we achieve a better and better approximation of our signal's magnitude against the frequency domain. The nfft dictates the number of sample points our resulting DFT (fx in this case) will have. It also follows from our analysis that numbers that are powers of 2 fit the algorithm and provide much better results. We can see this when we compare nfft=26 to all the other results we have achieved.

* + 1. Use ***ifft*** to find ***ix,*** theinverse DFT of each ***fx***, using the same ***nfft*** that produced ***fx***. Using the ***subplot*** option, ***plot*** ***ix*** versus *time*. Compare ***ix*** to ***x*** and draw a conclusion from your observation. This exercise shows the equivalence of the choice of ***nfft*** and zero padding.

  
With this graph we are able to see how ifft sampled based on the nfft to derive the original x function. In essence we are applying the algorithm to the nfft-point DFT.   
We begin with no zero padding, and so we are able to see one whole period of our wave form (as seen in the first image) and we follow with aggregation of samples and thus zero padding. The result is an increase in points and samples per period.   
NOTE: I changed the sinusoid to a cosine in this graph.

* 1. Compute ***fx*** for , , ***nfft***, using a window  {Rectangular, Hamming, Bartlett}. Compare the results in terms of main-lobe width and side-lobe height.

*Paste two figures and the MATLAB code here*: One figure should show windows versus time on the same plot using different colors, and another figure should show log-magnitude plots of the windows in the same plot using different colors. Use the ***legend*** option to indicate the window name. Choose a fixed amplitude value, e.g. when amplitude becomes 20 dB lower than the amplitude at 0 frequency, to define the main lobe width. Call that frequency fm . Choose a frequency that marks the beginning of the side band, e.g. when the amplitude reaches the first local maximum for f> fm . Applying these definitions to all three windows, *tabulate main-lobe width and side-lobe height for each window.*

%% d,e; beginning work on the the window portion.

%

clc,clear all,close all;

f0=1000;

fs=16\*f0;

ts=1/fs;

Td=1;

nfft=128;

strR='Rectangular';

strH='Hamming';

strB='Barlett';

t=0:ts:Td-ts;

tw=0:1/nfft:Td-1/nfft;

lt=length(t);

x=cos(2\*pi\*f0\*t)+cos(2\*pi\*3000\*t);

w=[window(@rectwin,nfft) window(@hamming,nfft) window(@bartlett,nfft) ];

w=w';

fw=[];

ww=[];

fxw=[];

fAbsXw=[];

a=length(x)-128;

z=zeros(1,a);

xw=[];

for k=1:3

ww(k,:)=[w(k,:) z];

xw(k,:)=ww(k,:).\*x;

fxw(k,:)=fft(xw(k,:));

fAbsXw(k,:)=abs(fxw(k,:));

end

wAbs=[];

for k=1:3

fw(k,:)=fft(w(k,:),nfft);

wAbs(k,:)=abs(fw(k,:));

end

f=(0:nfft/2-1)\*2\*pi/nfft;

figure(1),plot(tw,w),legend(strR,strH,strB); % plotting against time;

figure(2),plot(f,20\*log10(wAbs(:,1:length(f)))),legend(strR,strH,strB);

figure(3),plot(t(1,1:128),xw(:,1:length(tw))),legend(strR,strH,strB);

figure(4),plot(t,xw),legend(strR,strH,strB);

% fwx=(0:length(fAbsXw(1,:))/2-1)\*2\*pi\*1/length(fAbsXw(1,:));

fwx=(0:length(fAbsXw(1,:))/2-1)\*2\*pi\*ts;

figure(5),plot(fwx,fAbsXw(:,1:length(fwx))),legend(strR,strH,strB),title('mag(XW) vs OMEGA');

figure(5),plot(fwx,20\*log10(fAbsXw(:,1:length(fwx)))),legend(strR,strH,strB),title('mag(XW) dB vs OMEGA');

col='brgk';

zoom xon;

The Graphs below shows us the transform of the window times a sinusoidal signal x.

This signal had the same initial fundamental frequency as the first exercises (1000 Hz). Each point highlighted below defines the main lobe widths, or where the amplitude is 20dB lower than the main amplitude. It is possible to see how in the frequency domain, the rectangular window spans for a much broader frequency spectrum as opposed to the other windows.   
The second Graph show the x\*window in time domain. We can see the effect of each window to the signal in the time domain basis.  **

**

* 1. Repeat 1.d when *x*[n] is the sum of a 1 Hz and a 3 Hz. sinusoid.

Paste figures as in 1.d. and the MATLAB code here. Comment on the resolvability of the sinusoids from their frequency spectra. What is the effect of the windows on resolution? Resolution here means the ability to differentiate the two frequencies.   
  
Below are the images. The code is the same as above, except I defined x as follows: "x=cos(2\*pi\*f0\*t)+cos(2\*pi\*3000\*t);"   
It is possible to see that the windows perform a good job at isolating the frequencies that we wish to retain in our signal. However, Bartlett and Hamming windows perform a even better job than the rectangular window.





1. Exercises with EEG signals functions:

Download, read and plot the data file chb01\_01\_edfm.mat. Please indicate which one you chose to use. Observe the signals and all the channels in both the time domain and the frequency domain using the programs I provided for you. This step has no credit value but you should really do this exercise.

Save in variable x channel 7 of the data. For this exercise, you may clear the rest of the matrix.

* 1. Evaluate the ***fft*** of ***x*** for an appropriate value of ***nfft***. Check for interfering tones and use notch filters to remove them. Save the filtered signal in ***y***.

Paste your code here.

clear;close all;clc;

load('chb01\_02\_edfm.mat');

% sr=256;

x=val(7,:);

nfft=2^(fix(log2(length(x)))+2);

figure,plot(x),title('x vs sample');

clear val;

X=fft(x,nfft);

absX=abs(X);

% From the information file, we know that the

% sampling frequency is 256Hz.

% In addition, we also know that the duration is 10 minutes;(0:10);

fs=256;

ts=1/fs;

td=ts\*length(x); % the duration of the siganl equals to the period of each sample times the total number of samples in the signal;

t=0:ts:td-ts;

figure,plot(t,x),title('x vs time t');

f=(0:nfft/2-1)\*fs/nfft;

figure, plot(f,absX(1,1:nfft/2)),ylabel('mag(X)'),xlabel('frequency f'),title('mag of X vs f without filtering;');

om=2\*pi\*f/fs;

figure, plot(om,20\*log10(absX(1,1:nfft/2))),ylabel('mag(X) dB'),xlabel('frequency f'),title('mag (x) dB vs Omega Without Filtering');

% After using the spectral tip to determine the frequencies to take away

% from our signal, we will define them here an begin the filtering.

w1=16;w2=32;w3=44;w4=48;w5=60;w6=64;w7=76;w8=80;w9=96;w10=112;

w=[w1 w2 w3 w4 w5 w6 w7 w8 w9 w10];

w=w/fs\*2\*pi;

% after we have all of our frequencies decided, we begin by devising our

% filters based on that information;

r=0.99;

H=[zeros(10,nfft/2)];

w0=[zeros(1,10)];

a=[zeros(10,3)];

b=[zeros(10,3)];

for k=1:10

w0=w(1,k);

b(k,:)=[1 -2\*cos(w0) 1];

a(k,:)=[1 -2\*r\*cos(w0) r^2];

H(k,:)=freqz(b(k,:),a(k,:),nfft/2);% H represents our filter function;

end

figure,

hold on,

grid minor,

for k=1:10

plot(om,20\*log10(abs(H(k,:)))) ;

end

plot(om,20\*log10(absX(1,1:nfft/2)),'-r'),

title('dB of the filter function and magX');

clear y k;

y=x;

for k=1:10

y=filter(b(k,:),a(k,:),y);

%k;

end

Y=fft(y,nfft);

magY=abs(Y(1:nfft/2));

figure,

plot(f,magY),

title('magY vs f');

figure,

plot(f,20\*log10(magY)),title('dB of the magnitude of Y');

zoom xon;

   
After implementing our filter to our x function, this is what the result Y looks like: 

We can actually note each of the suppressed bands by inspecting the image.

* 1. Using a Hamming window, and a frame duration of 3 seconds, an overlap of 60 %, divide x and y into frames, and plot their spectrograms.

Paste your code and the spectrograms here.

* 1. You are given a crude set of 6 filters in FBMod.m
     1. Input ***y*** to each of the 6 band-pass filters. Call the output ***z***, where ***z*** is a matrix of 6 columns for each filter output***.*** Plot ***z*** using subplots.

Paste your code and the plots figure here.

* + 1. Add the 6 columns of ***z*** to obtain ***sz***. Using subplots, plot ***sz*** and ***y.*** Compare ***sz*** and ***y*** in terms of wave shape and delay. Are they nearly equal? By how many samples is ***sz*** delayed relative to ***y***? Can you account for the delay?

Paste the figure and the code here. Answer the questions.

* + 1. Evaluate the DFT of ***z*** and use subplot to plot the magnitude of each column. Do they each occupy a different frequency band? State the approximate frequency range of each band in Hz.

Paste figure and code here. Answer the questions.

* + 1. Using ***frames.m*** divide each column of ***z*** into 2 second windows overlapping by 95%. Find the energy of each frame divided by the length of the frame. Save the energy computations in a matrix ***ez*** with six columns such that each column is the energy over time of each column of ***z***. Plot columns of ***ez***.

Paste your code and the figure of plots here.

* + 1. Comment on your observation relative to seizure detection.

Where are the seizures? Why do you think they are seizures? Please refer to the cited papers and the dissertation of Ali Hossam Shoeb from the Physionet web site in the description of the seizures.